

Exhibit D to the
Declaration of Imran A. Khaliq In Support
Of Visto's Opening Claim Construction
Brief Under P.R. 4-5(a)

(12) **United States Patent**
Mendez et al.

(10) **Patent No.:** **US 6,708,221 B1**
 (45) **Date of Patent:** **Mar. 16, 2004**

(54) **SYSTEM AND METHOD FOR GLOBALLY AND SECURELY ACCESSING UNIFIED INFORMATION IN A COMPUTER NETWORK**

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,388,255 A * 2/1995 Pytlik et al. 707/4
 5,544,320 A * 8/1996 Konrad 709/203

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

CA	2210763	1/1999
EP	0801478	10/1997
WO	WO 99/05620	2/1999
WO	WO 99/45451	9/1999

OTHER PUBLICATIONS

US 5,373,559, 12/1994, Kaufman et al. (withdrawn)

(List continued on next page.)

Primary Examiner—Mehmet B. Geckil

(74) *Attorney, Agent, or Firm*—Squire, Sanders & Dempsey L.L.P.

(57) **ABSTRACT**

A client stores a first set of workspace data, and is coupled via a computer network to a global server. The client may be configured to synchronize portions of the first set of workspace data with the global server, which stores independently modifiable copies of the portions. The global server may also store workspace data which is not downloaded from the client, and thus stores a second set of workspace data. The global server may be configured to identify and authenticate a user seeking global server access from a remote terminal, and is configured to provide access to the first set or to the second set. Further, services may be stored anywhere in the computer network. The global server may be configured to provide the user with access to the services. The system may further include a synchronization-start module at the client site (which may be protected by a firewall) that initiates interconnection and synchronization with the global server when predetermined criteria have been satisfied.

(75) **Inventors:** **Daniel J. Mendez**, Menlo Park, CA (US); **Mark D. Riggins**, Mercer Island, WA (US); **Prasad Wagle**, Santa Clara, CA (US); **Hong Q. Bui**, Cupertino, CA (US); **Mason Ng**, Mountain View, CA (US); **Sean Michael Quinlan**, San Francisco, CA (US); **Christine C. Ying**, Foster City, CA (US); **Christopher R. Zuleeg**, San Jose, CA (US); **David J. Cowan**, Menlo Park, CA (US); **Joanna A. Aptekar-Strober**, Menlo Park, CA (US); **R. Stanley Bailes**, San Jose, CA (US)

(73) **Assignee:** **Visto Corporation**, Redwood Shores, CA (US)

(*) **Notice:** Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) **Appl. No.:** **09/666,877**

(22) **Filed:** **Sep. 20, 2000**

Related U.S. Application Data

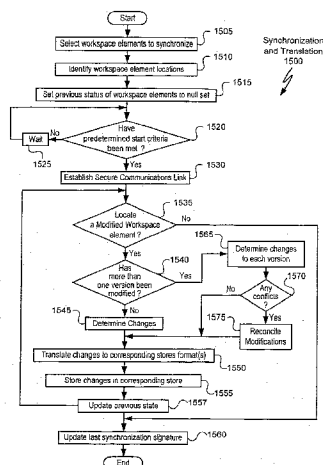
(63) Continuation of application No. 08/903,118, filed on Jul. 30, 1997, and a continuation-in-part of application No. 08/865,075, filed on May 29, 1997, now Pat. No. 6,023,708, and a continuation-in-part of application No. 08/835,997, filed on Apr. 11, 1997, now Pat. No. 6,085,192, and a continuation-in-part of application No. 08/841,950, filed on Apr. 8, 1997, which is a continuation-in-part of application No. 08/766,307, filed on Dec. 13, 1996, now Pat. No. 6,131,116.

(51) **Int. Cl.**⁷ **G06F 15/16**

(52) **U.S. Cl.** **709/248; 709/204**

(58) **Field of Search** 709/203, 219, 709/248, 245, 204, 205

14 Claims, 15 Drawing Sheets



U.S. PATENT DOCUMENTS

5,588,132	A	*	12/1996	Cardoza	711/148	
5,752,246	A		5/1998	Rogers et al.			
5,764,902	A	*	6/1998	Rothrock	345/753	
5,812,773	A	*	9/1998	Norin	345/753	
5,835,601	A	*	11/1998	Shimbo et al.	709/201	
5,862,346	A	*	1/1999	Kley et al.	380/29	
5,878,230	A		3/1999	Weber et al.			
5,924,103	A	*	7/1999	Ahmed et al.	707/201	
5,928,329	A		7/1999	Clark et al.			
5,943,676	A	*	8/1999	Boothby	707/201	
5,974,238	A	*	10/1999	Chase, Jr.	709/248	
5,999,932	A		12/1999	Paul			
5,999,947	A	*	12/1999	Zollinger et al.	707/203	
6,020,885	A	*	2/2000	Honda	345/757	
6,021,427	A		2/2000	Spagna et al.			
6,023,700	A		2/2000	Owens et al.			
6,023,708	A	*	2/2000	Mendez et al.	707/203	
6,034,621	A		3/2000	Kaufman			
6,073,165	A		6/2000	Narasimhan et al.			
6,094,477	A		7/2000	Nada et al.			
6,108,709	A		8/2000	Shinomura et al.			
6,118,856	A		9/2000	Paarsmarkt et al.			
6,125,281	A		9/2000	Wells et al.			
6,131,096	A	*	10/2000	Ng et al.	707/10	
6,131,116	A	*	10/2000	Riggins et al.	709/219	
6,138,146	A		10/2000	Moon et al.			
6,212,529	B1	*	4/2001	Boothby et al.	707/201	
6,249,805	B1		6/2001	Fleming, III			
6,295,541	B1		9/2001	Bodnar et al.			
6,343,313	B1	*	1/2002	Salesky et al.	345/751	
6,510,455	B1		1/2003	Chen et al.			

OTHER PUBLICATIONS

Kohl, John T., et al.; "The Evolution of the *Kerberos* Authentication Service"; 1991; pp. 1-15; This paper is a revision of a paper presented at the Spring 1991 EurOpen Conference in Tromso, Norway.

Adams, Charlotte; "Multilevel Secure Networking Charges Ahead"; Federal Computer Week; Apr. 12, 1993; 5 pages.

Jaeger, Trent and Atul Prakash; "Implementation of a Discretionary Access Control Model for Script-based Systems"; IEEE Jun. 1995; 15 pages.

Research Disclosure; "Provide Auto-Forwarding Based on Criteria Selected by the User"; Oct. 1, 1989; 1 page; No. 306; Kenneth Mason Publications; XP000085405; ISSN; 0374-4353.

* cited by examiner

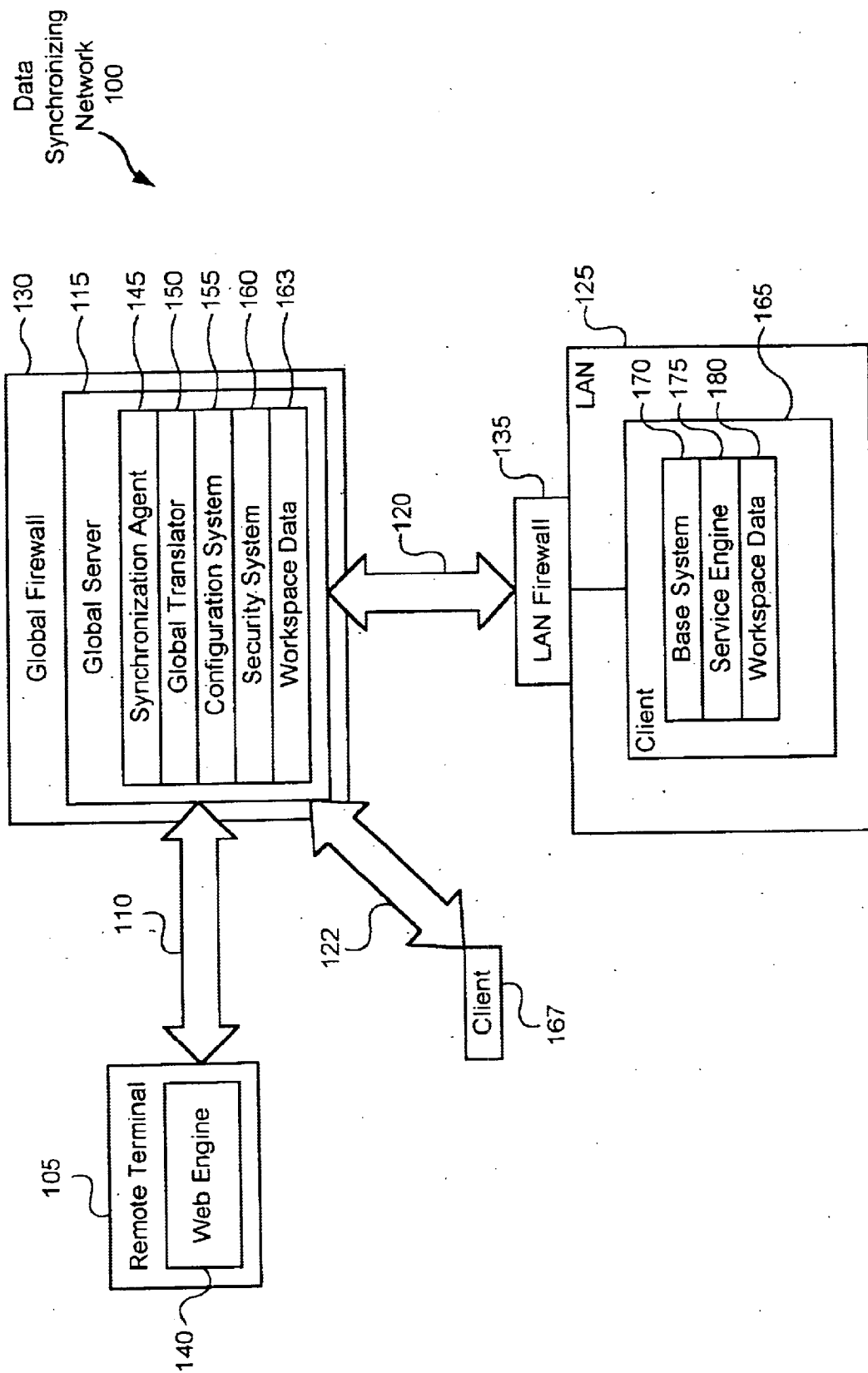


FIG. 1

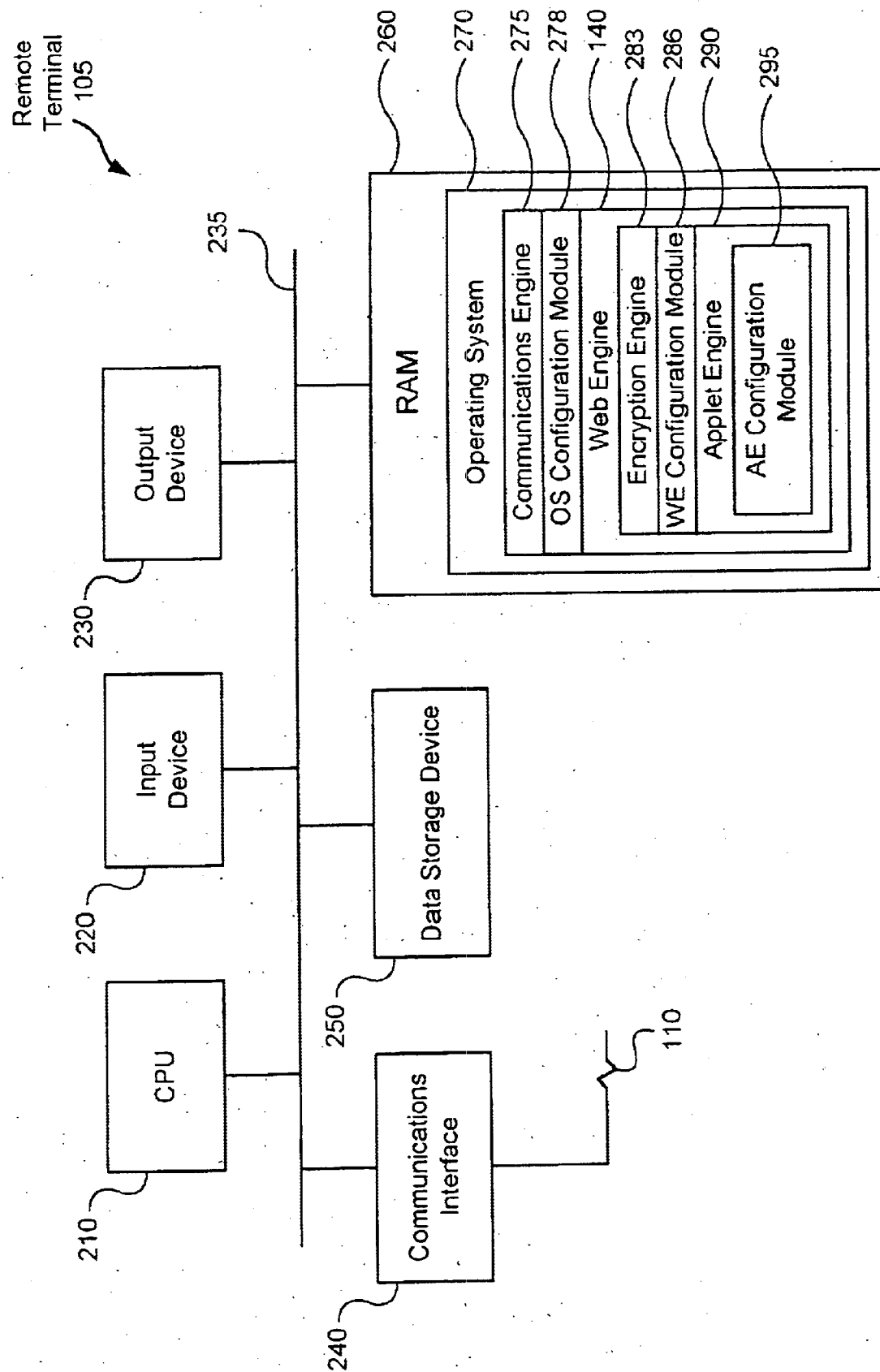


FIG. 2

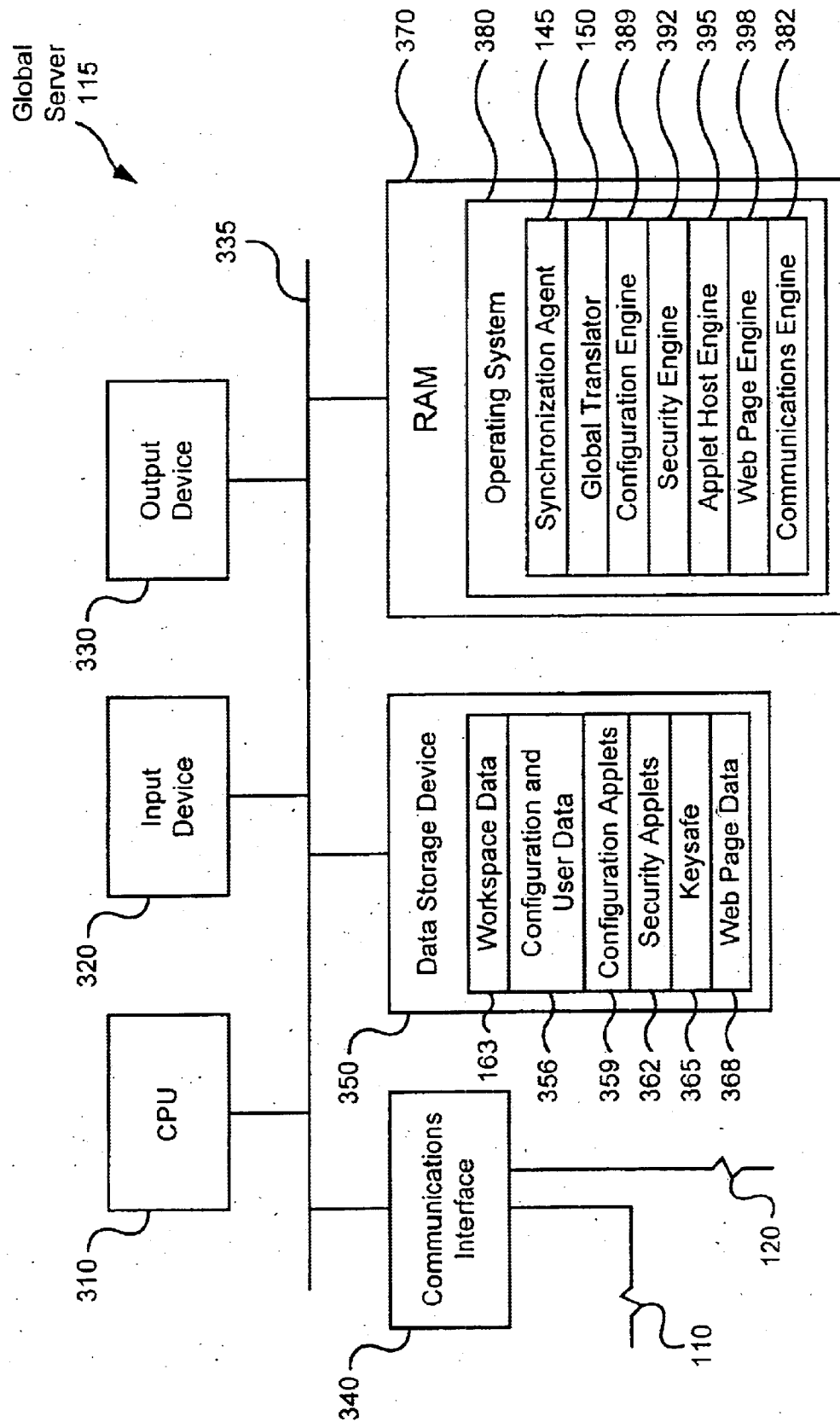


FIG. 3

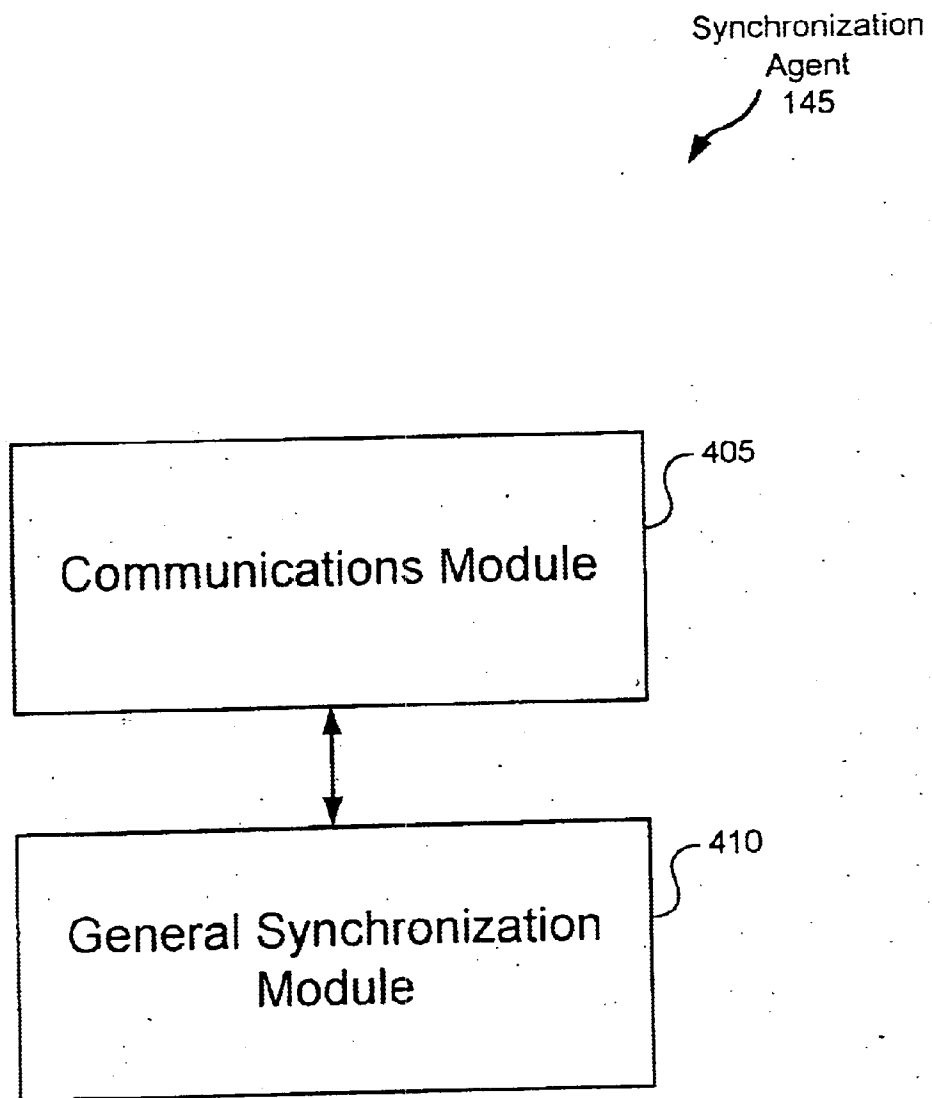
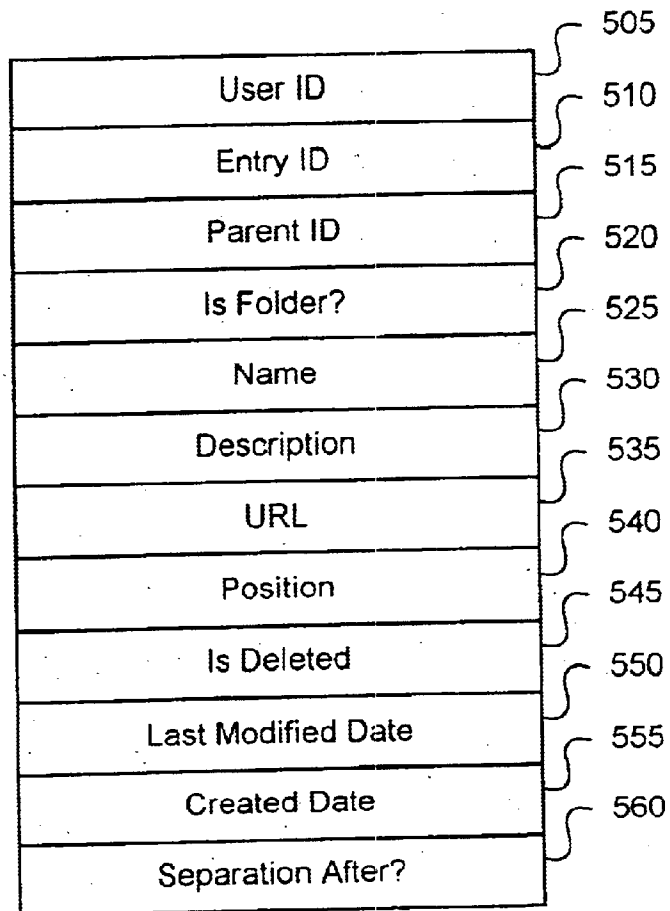


FIG. 4

Global Format
Bookmark
(example)
500



User ID	505
Entry ID	510
Parent ID	515
Is Folder?	520
Name	525
Description	530
URL	535
Position	540
Is Deleted	545
Last Modified Date	550
Created Date	555
Separation After?	560

FIG. 5

Configuration
and user data

356

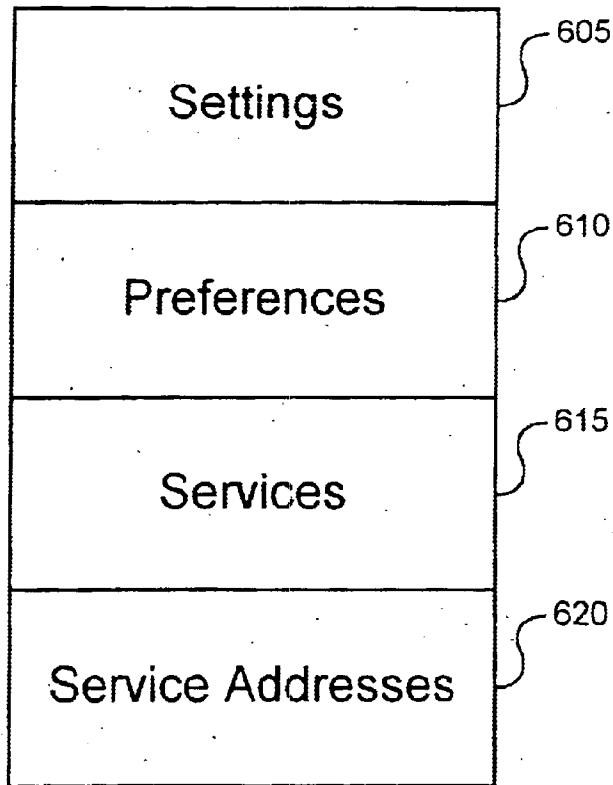


FIG. 6

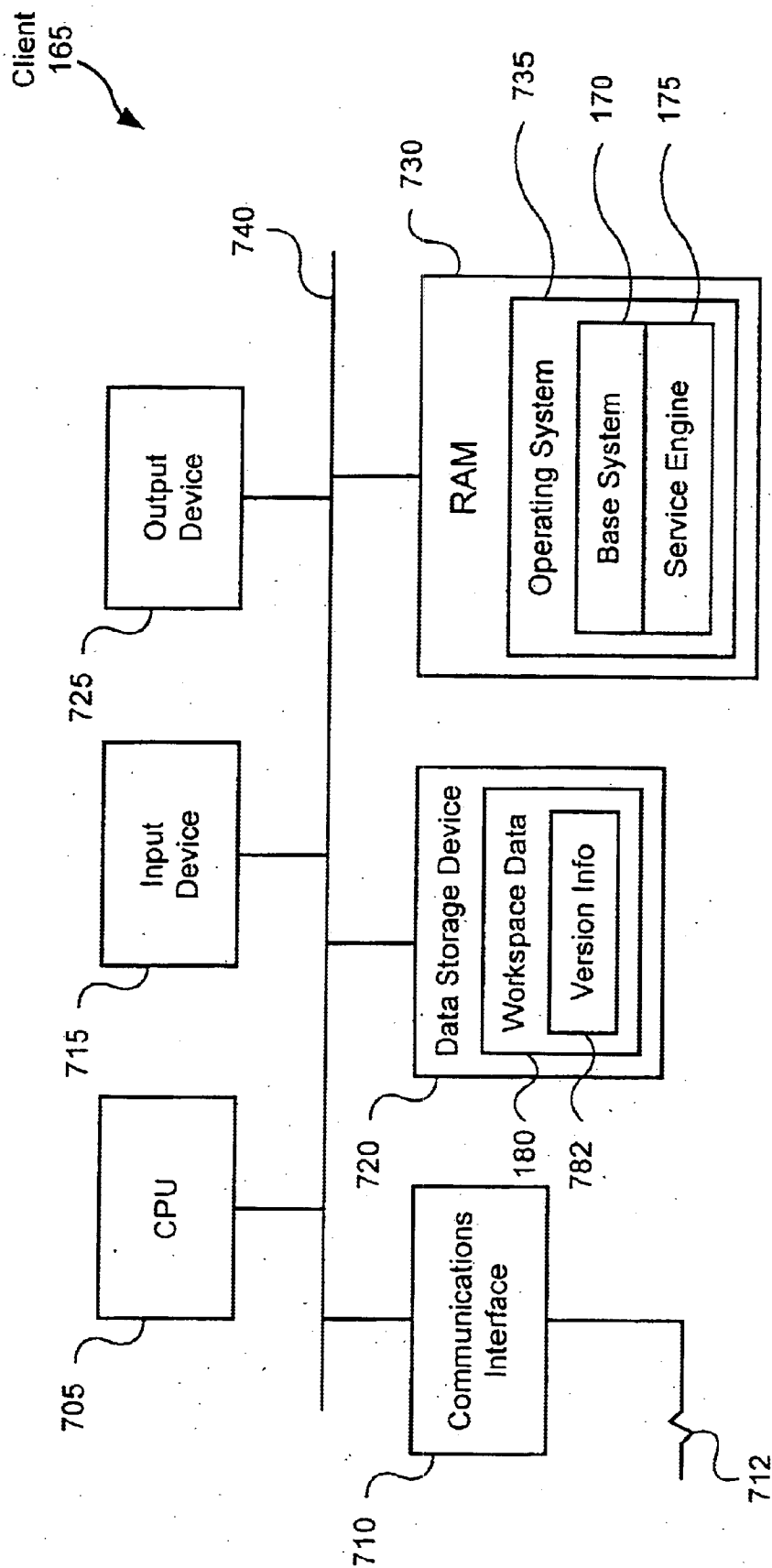


FIG. 7

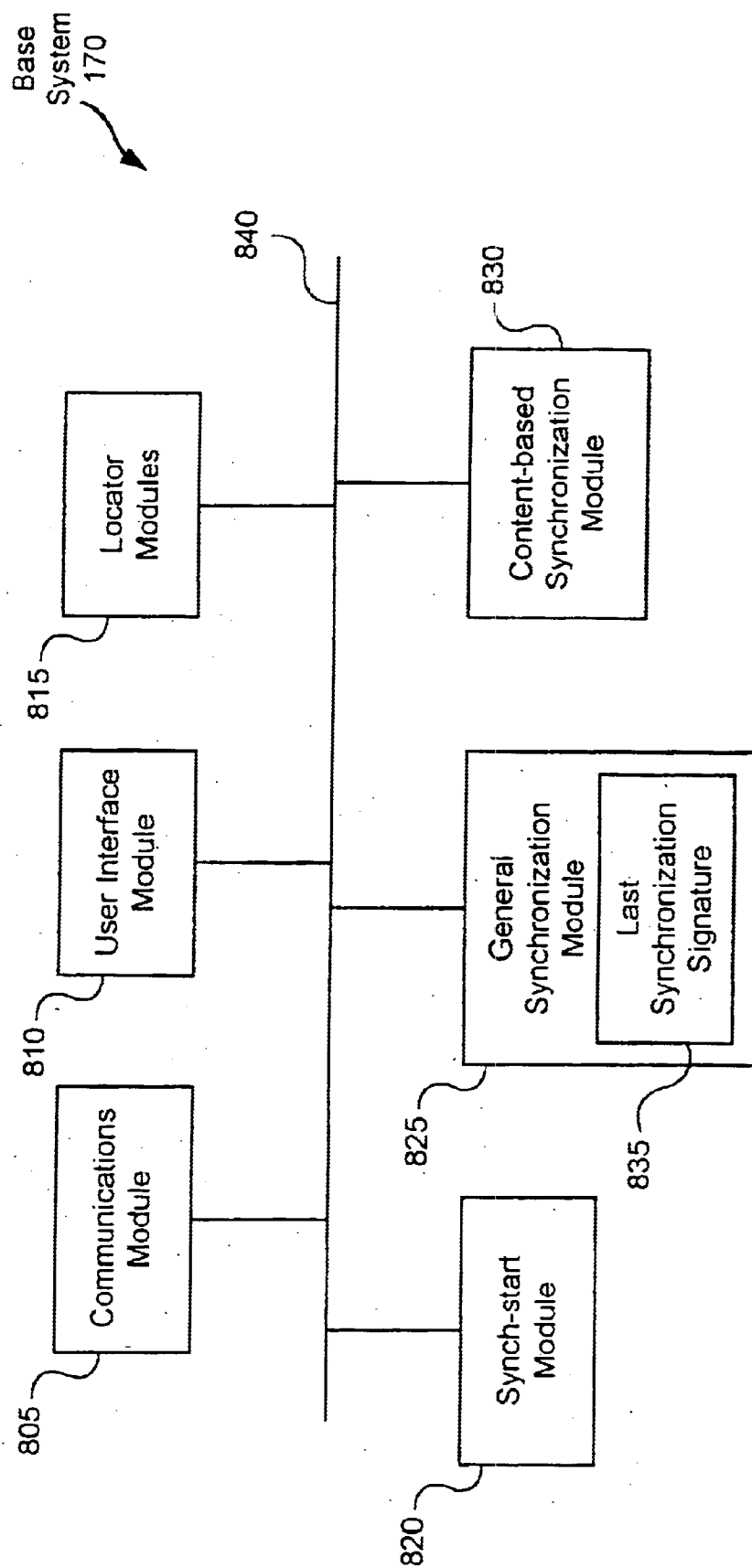


FIG. 8

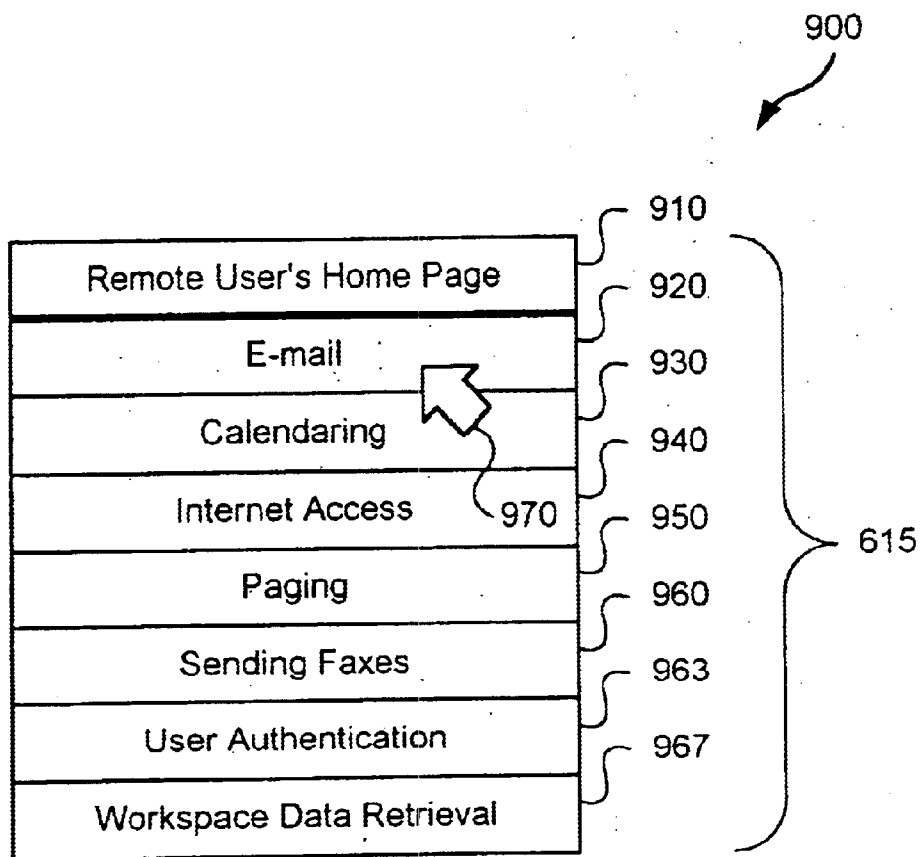


FIG. 9

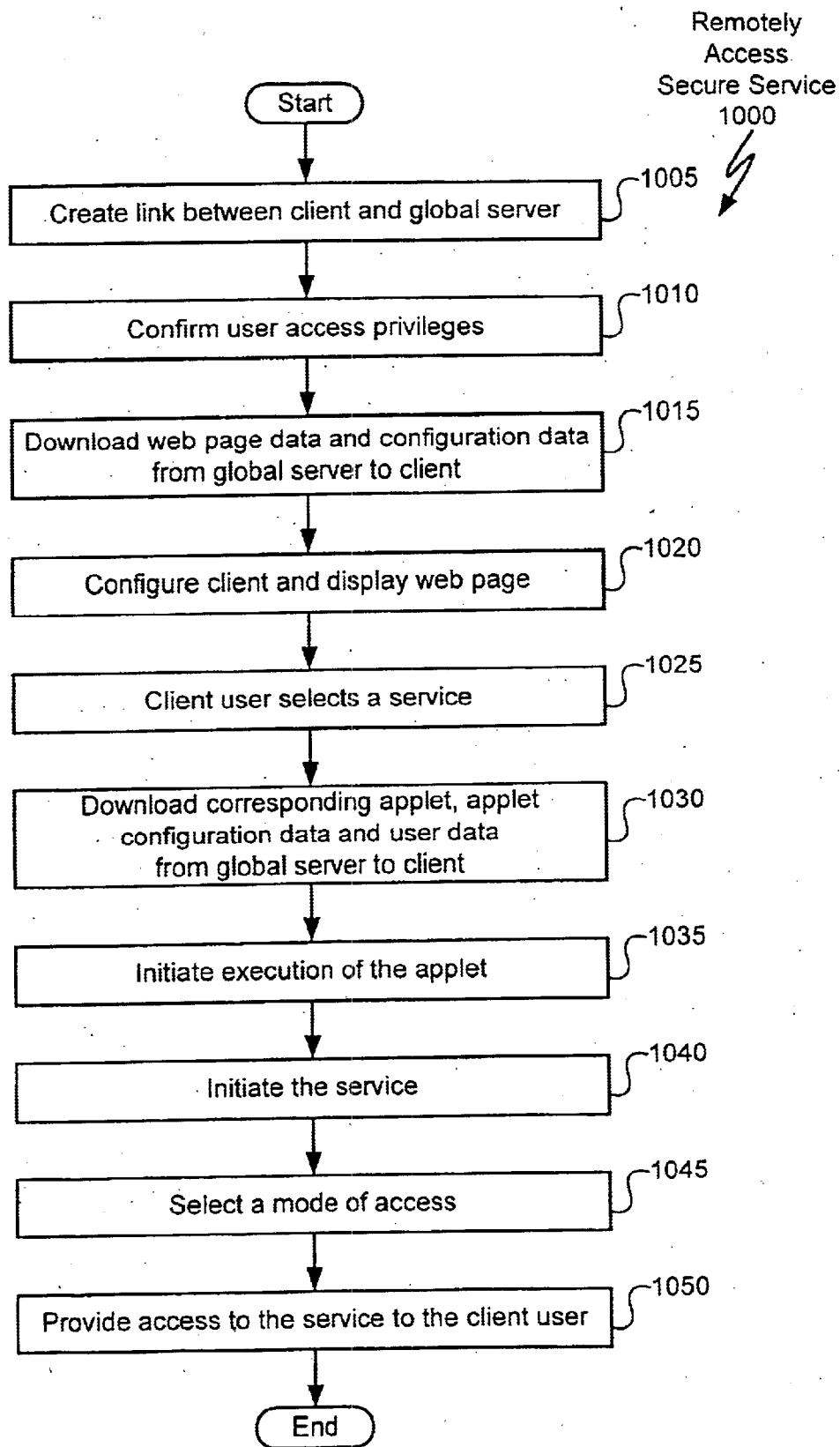


FIG. 10

Create link
between client
& server
1005

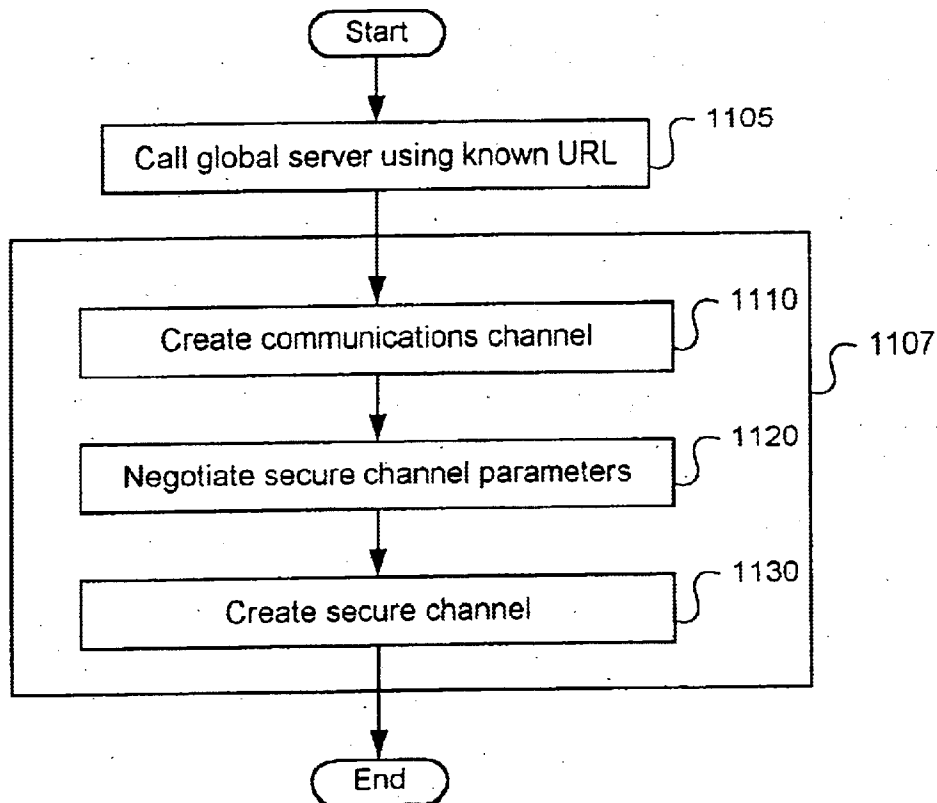
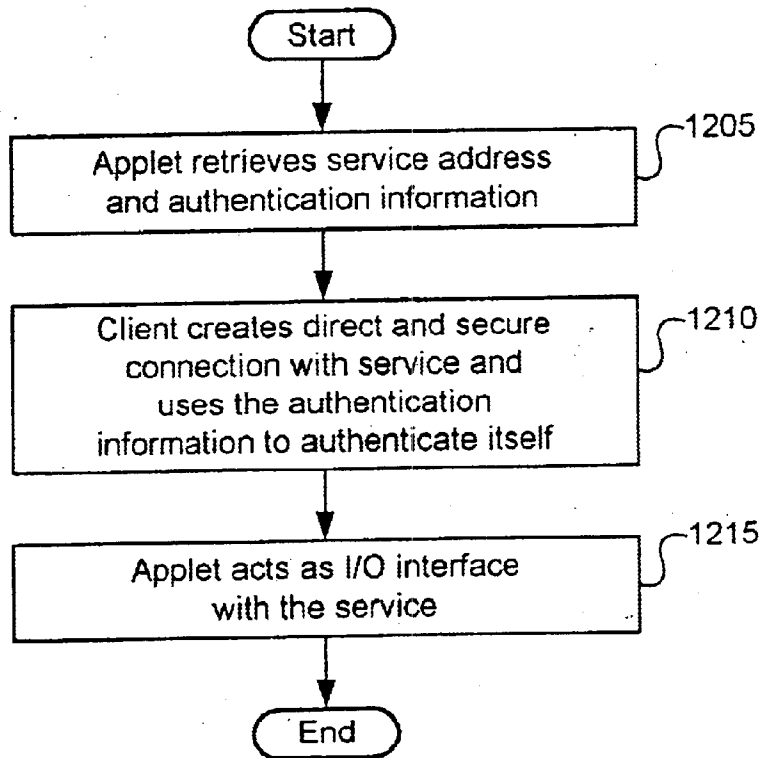


FIG. 11

Method of
accessing service
1050a



(Redirect)

FIG. 12

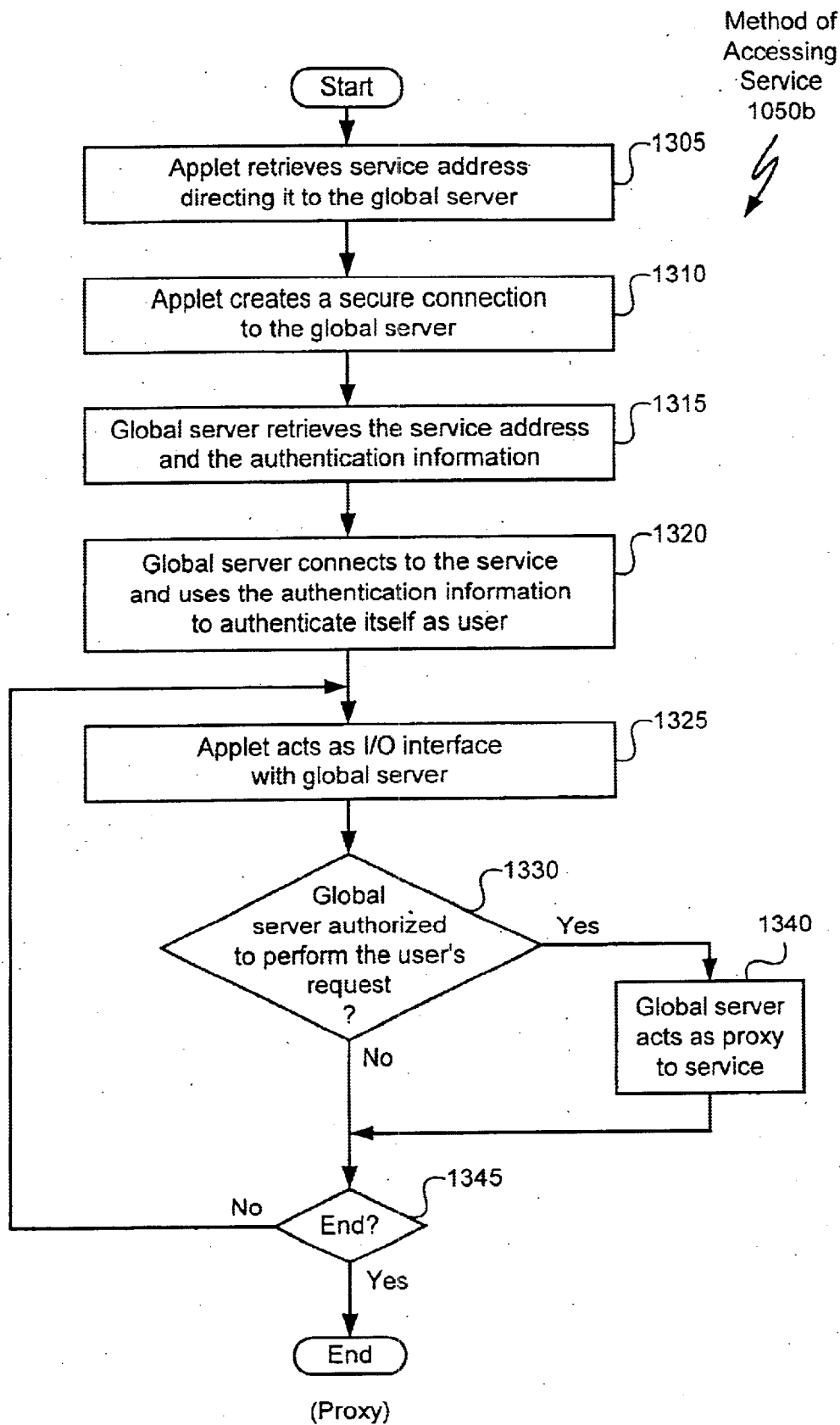


FIG. 13

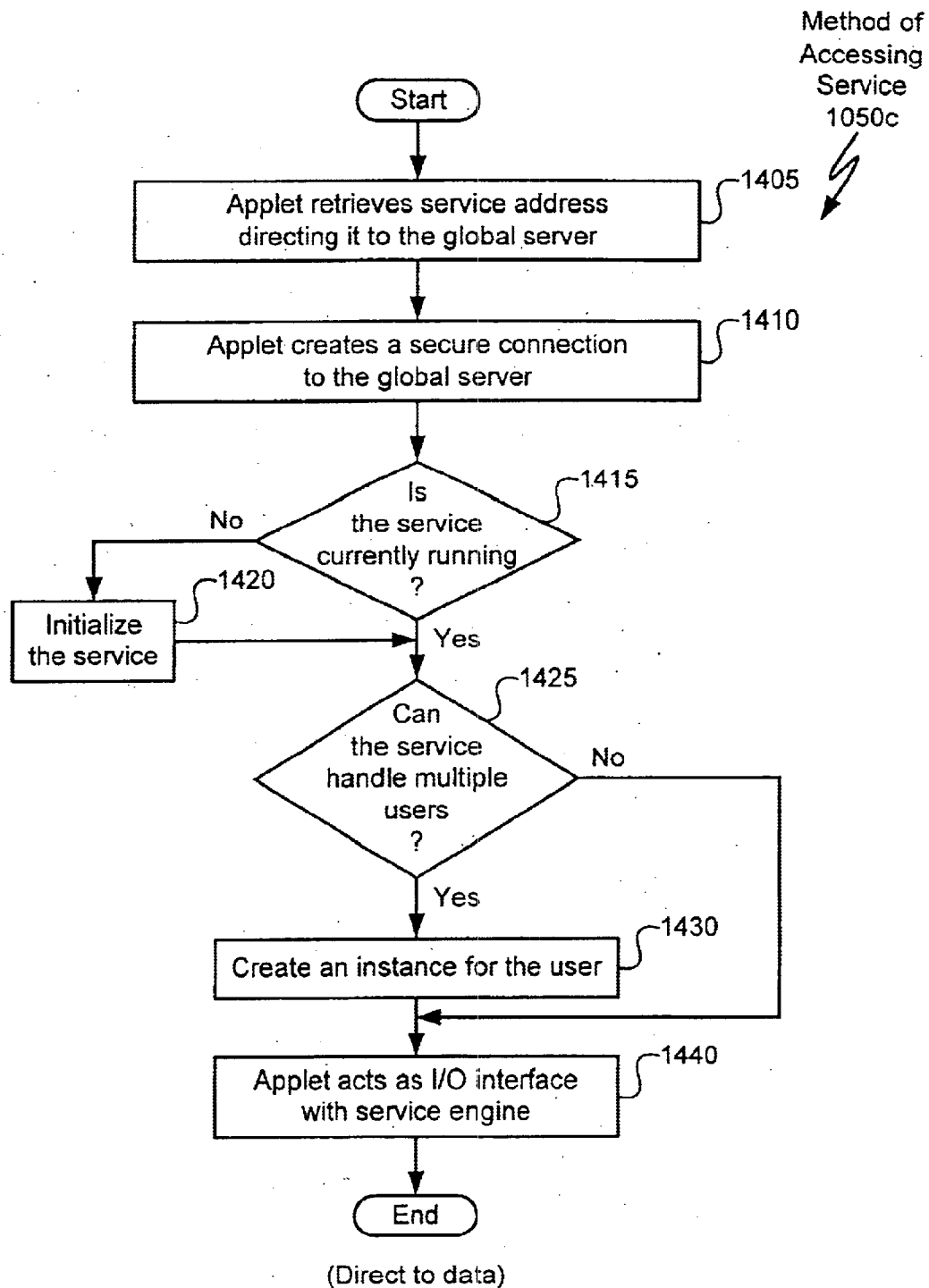


FIG. 14

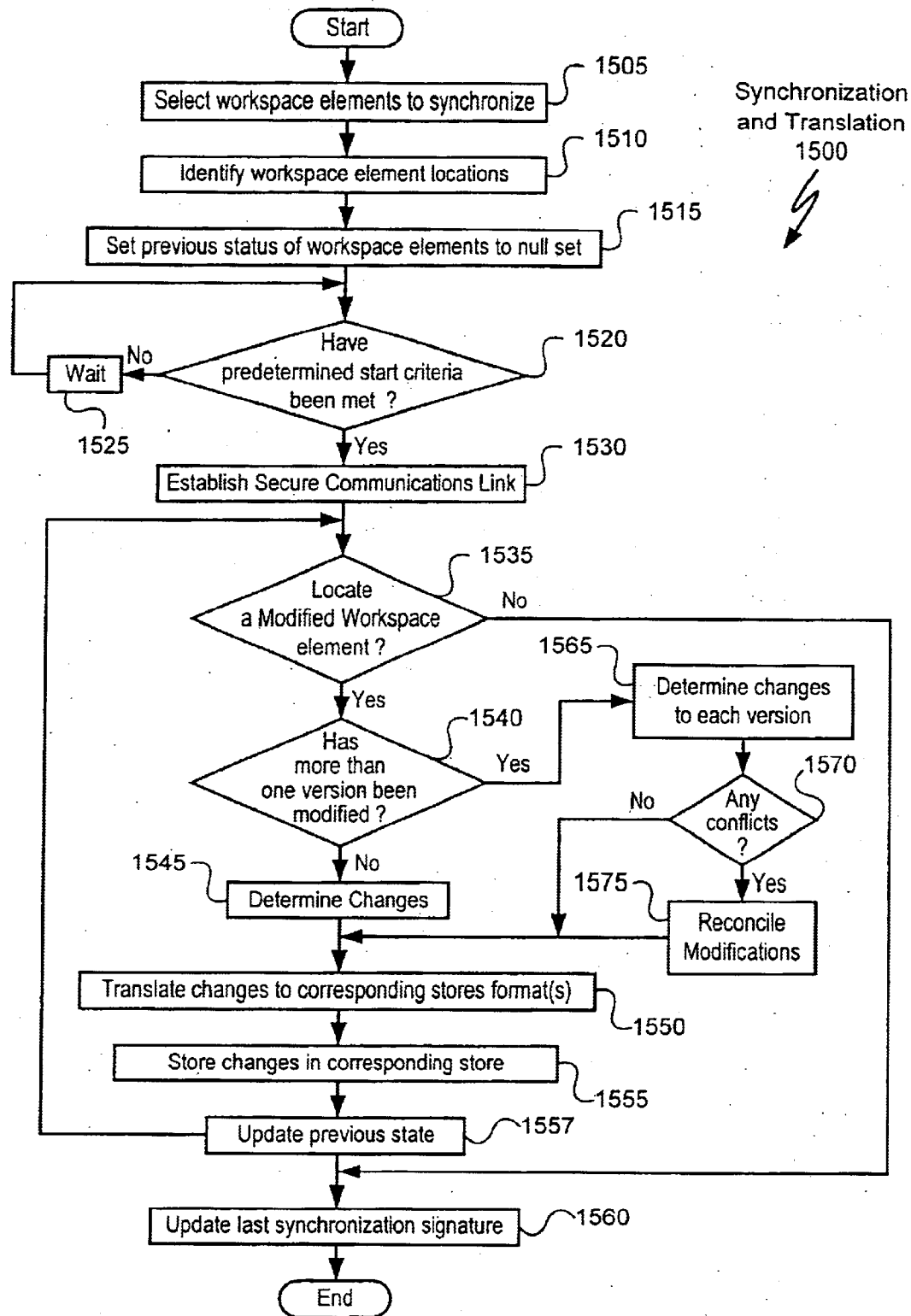


FIG. 15

1

SYSTEM AND METHOD FOR GLOBALLY AND SECURELY ACCESSING UNIFIED INFORMATION IN A COMPUTER NETWORK

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation and claims priority to and incorporates by reference U.S. patent application Ser. No. 08/903,118, entitled "System and Method For Globally and Securely Accessing Unified Information in a Computer Network" filed Jul. 30, 1997 of Daniel J. Mendez, Mark D. Riggins, Prasad Wagle, Hong Q. Bui, Mason Ng, Sean Michael Quinlan, Christine C. Ying, Christopher R. Zuleeg, David J. Cowan, Joanna A. Aptekar-Strober and R. Stanley Bailes, which is a continuation-in-part of U.S. patent application Ser. No. 08/766,307, entitled "System and Method for Globally Accessing Computer Services," filed on Dec. 13, 1996 now U.S. Pat. No. 6,131,116 by inventors Mark D. Riggins, R. Stanley Bailes, Hong Q. Bui, David J. Cowan, Daniel J. Mendez, Mason Ng, Sean Michael Quinlan, Prasad Wagle, Christine C. Ying, Christopher R. Zuleeg and Joanna A. Aptekar-Strober; and of co-pending U.S. patent application Ser. No. 08/841,950 entitled "System and Method for Enabling Secure Access to Services in a Computer Network," filed on Apr. 8, 1997 by inventor Mark Riggins; and of U.S. patent application Ser. No. 08/835,997 entitled "System and Method for Securely Synchronizing Multiple Copies of a Workspace Element in a Network," filed on Apr. 11, 1997 now U.S. Pat. No. 6,085,192, by inventors Daniel J. Mendez, Mark J. Riggins, Prasad Wagle and Christine C. Ying; and of U.S. patent application Ser. No. 08/865,075 entitled "System and Method for Using a Global Translator to Synchronize Workspace Elements Across a Network," filed on May 29, 1997 now U.S. Pat. No. 6,023,708 by inventors Daniel J. Mendez, Mark D. Riggins, Prasad Wagle and Christine C. Ying. These applications have been commonly assigned to RoamPage, Inc. and are incorporated herein by reference as if copied verbatim hereafter. Benefit of the earlier filing dates is claimed on all common subject matter.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates generally to computer networks, and more particularly provides a system and method for globally and securely accessing unified information in a computer network.

2. Description of the Background Art

The internet currently interconnects about 100,000 computer networks and several million computers. Each of these computers stores numerous application programs for providing numerous services, such as generating, sending and receiving e-mail, accessing World Wide Web sites, generating and receiving facsimile documents, storing and retrieving data, etc.

A roaming user, i.e., a user who travels and accesses a workstation remotely, is faced with several problems. Program designers have developed communication techniques for enabling the roaming user to establish a communications link and to download needed information and needed service application programs from the remote workstation to a local computer. Using these techniques, the roaming user can manipulate the data on the remote workstation and, when finished, can upload the manipulated data back from the remote workstation to the local computer. However, slow

2

computers and slow communication channels make downloading large files and programs a time-consuming process. Further, downloading files and programs across insecure channels severely threatens the integrity and confidentiality of the downloaded data.

Data consistency is also a significant concern for the roaming user. For example, when maintaining multiple independently modifiable copies of a document, a user risks using an outdated version. By the time the user notices an inconsistency, interparty miscommunication or data loss may have already resulted. The user must then spend more time attempting to reconcile the inconsistent versions and addressing any miscommunications.

The problem of data inconsistency is exacerbated when multiple copies of a document are maintained at different network locations. For example, due to network security systems such as conventional firewall technology, a user may have access only to a particular one of these network locations. Without access to the other sites, the user cannot confirm that the version on the accessible site is the most recent draft.

Data consistency problems may also arise when using application programs from different vendors. For example, the Netscape Navigator™ web engine and the Internet Explorer™ web engine each store bookmarks for quick reference to interesting web sites. However, since each web engine uses different formats and stores bookmarks in different files, the bookmarks are not interchangeable. In addition, one web engine may store a needed bookmark, and the other may not. A user who, for example, runs the Internet Explorer™ web engine at home and runs the Netscape Navigator™ web engine at work risks having inconsistent bookmarks at each location.

Therefore, a system and method are needed to enable multiple users to access computer services remotely without consuming excessive user time, without severely threatening the integrity and confidentiality of the data, and without compromising data consistency.

SUMMARY OF THE INVENTION

The present invention provides a system and methods for providing global and secure access to services and to unified (synchronized) workspace elements in a computer network. A user can gain access to a global server using any terminal, which is connected via a computer network such as the Internet to the global server and which is enabled with a web engine.

A client stores a first set of workspace data, and is coupled via a computer network to a global server. The client is configured to synchronize selected portions of the first set of workspace data (comprising workspace elements) with the global server, which stores independently modifiable copies of the selected portions. The global server may also store workspace data not received from the client, such as e-mail sent directly to the global server. Accordingly, the global server stores a second set of workspace data. The global server is configured to identify and authenticate a user attempting to access it from a remote terminal, and is configured to provide access based on the client configuration either to the first set of workspace data stored on the client or to the second set of workspace data stored on the global server. It will be appreciated that the global server can manage multiple clients and can synchronize workspace data between clients.

Service engines for managing services such as e-mail management, accessing bookmarks, calendaring, network

3

access, etc. may be stored anywhere in the computer network, including on the client, on the global server or on any other computer. The global server is configured to provide the user with access to services, which based on level of authentication management or user preferences may include only a subset of available services. Upon receiving a service request from the client, the global server sends configuration information to enable access to the service.

Each client includes a base system and the global server includes a synchronization agent. The base system and synchronization agent automatically establish a secure connection therebetween and synchronize the selected portions of the first set of workspace data stored on the client and the second set of workspace data stored on the global server. The base system operates on the client and examines the selected portions to determine whether any workspace elements have been modified since last synchronization. The synchronization agent operates on the global server and informs the base system whether any of the workspace elements in the second set have been modified. Modified version may then be exchanged so that an updated set of workspace elements may be stored at both locations, and so that the remote user can access an updated database. If a conflict exists between two versions, the base system then performs a responsive action such as examining content and generating a preferred version, which may be stored at both locations. The system may further include a synchronization-start module at the client site (which may be protected by a firewall) that initiates interconnection and synchronization when predetermined criteria have been satisfied.

A method of the present invention includes establishing a communications link between the client and the global server. The method includes establishing a communications link between the client and a service based upon user requests. The method receives configuration data and uses the configuration data to configure the client components such as the operating system, the web engine and other components. Configuring client components enables the client to communicate with the service and provides a user-and-service-specific user interface on the client. Establishing a communications link may also include confirming access privileges.

Another method uses a global translator to synchronize workspace elements. The method includes the steps of selecting workspace elements for synchronization, establishing a communications link between a client and a global server, examining version information for each of the workspace elements on the client and on the global server to determine workspace elements which have been modified since last synchronization. The method continues by comparing the corresponding versions and performing a responsive action. Responsive actions may include storing the preferred version at both stores or reconciling the versions using content-based analysis.

The system and methods of the present invention advantageously provide a secure globally accessible third party, i.e. the global server. The system and methods provide a secure technique for enabling a user to access the global server and thus workspace data remotely and securely. Because of the global firewall and the identification and security services performed by the global server, corporations can store relatively secret information on the global server for use by authorized clients. Yet, the present invention also enables corporations to maintain only a portion of their secret information on the global server, so that there would be only limited loss should the global server be compromised. Further, the global server may advantageously

4

act as a client proxy for controlling access to services, logging use of keys and logging access of resources.

A client user who maintains a work site, a home site, an off-site and the global server site can securely synchronize the workspace data or portions thereof among all four sites. Further, the predetermined criteria (which control when the synchronization-start module initiates synchronization) may be set so that the general synchronization module synchronizes the workspace data upon user request, at predetermined times during the day such as while the user is commuting, or after a predetermined user action such as user log-off or user log-on. Because the system and method operate over the Internet, the system is accessible using any connected terminal having a web engine such as an internet-enabled smart phone, television settop (e.g., web TV), etc. and is accessible over any distance. Since the system and method include format translation, merging of workspace elements between different application programs and different platforms is possible. Further, because synchronization is initiated from within the firewall, the typical firewall, which prevents in-bound communications and only some protocols of out-bound communications, does not act as an impediment to workspace element synchronization.

Further, a roaming user may be enabled to access workspace data from the global server or may be enabled to access a service for accessing workspace data from a client. For example, a user may prefer not to store personal information on the global server but may prefer to have remote access to the information. Further, the user may prefer to store highly confidential workspace elements on the client at work as added security should the global server be compromised.

The present invention may further benefit the roaming user who needs emergency access to information. The roaming user may request a Management Information Systems (MIS) director controlling the client to provide the global server with the proper keys to enable access to the information on the client. If only temporary access is desired, the keys can then be later destroyed either automatically or upon request. Alternatively, the MIS director may select the needed information as workspace elements to be synchronized and may request immediate synchronization with the global server. Accordingly, the global server and the client can synchronize the needed information, and the user can access the information from the global server after it has completed synchronization.

The present invention also enables the system and methods to synchronize keys, available services and corresponding service addresses to update accessibility of workspace data and services. For example, if the user of a client accesses a site on the Internet which requires a digital certificate and the user obtains the certificate, the system and methods of the present invention may synchronize this newly obtained certificate with the keys stored on the global server. Thus, the user need not contact the global server to provide it with the information. The synchronization means will synchronize the information automatically.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram illustrating a secure data-synchronizing remotely accessible network in accordance with the present invention;

FIG. 2 is a block diagram illustrating details of a FIG. 1 remote terminal;

FIG. 3 is a block diagram illustrating details of a FIG. 1 global server;

5

FIG. 4 is a block diagram illustrating details of a FIG. 1 synchronization agent;

FIG. 5 is a graphical representation of an example bookmark in global format;

FIG. 6 is a graphical representation of the FIG. 3 configuration data;

FIG. 7 is a block diagram illustrating the details of a FIG. 1 client;

FIG. 8 is a block diagram illustrating the details of a FIG. 1 base system;

FIG. 9 illustrates an example services list;

FIG. 10 is a flowchart illustrating a method for remotely accessing a secure server;

FIG. 11 is a flowchart illustrating details of the FIG. 10 step of creating a link between a client and global server;

FIG. 12 is a flowchart illustrating details of the FIG. 10 step of providing access to a service in a first embodiment;

FIG. 13 is a flowchart illustrating details of the FIG. 10 step of providing access to a service in a second embodiment;

FIG. 14 is a flowchart illustrating details of the FIG. 10 step of providing access to a service in a third embodiment; and

FIG. 15 is a flowchart illustrating a method for synchronizing multiple copies of a workspace element over a secure network.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

FIG. 1 is a block diagram illustrating a network 100, comprising a first site such as a remote computer terminal 105 coupled via a communications channel 110 to a global server 115. The global server 115 is in turn coupled via a communications channel 120 to a second site such as a Local Area Network (LAN) 125 and via a communications channel 122 to a third site such as client 167. Communications channel 110, communications channel 120 and communications channel 122 may be referred to as components of a computer network such as the Internet. The global server 115 is protected by a global firewall 130, and the LAN 125 is protected by a LAN firewall 135.

The LAN 125 comprises a client 165, which includes a base system 170 for synchronizing workspace data 180 (e-mail data, file data, calendar data, user data, etc.) with the global server 115 and may include a service engine 175 for providing computer services such as scheduling, e-mail, paging, word-processing or the like. Those skilled in the art will recognize that workspace data 180 may include other types of data such as application programs. It will be further appreciated that workspace data 180 may each be divided into workspace elements, wherein each workspace element may be identified by particular version information 782 (FIG. 7). For example, each e-mail, file, calendar, etc. may be referred to as "a workspace element in workspace data." For simplicity, each workspace element on the client 165 is referred to herein as being stored in format A. It will be further appreciated that the workspace data 180 or portions thereof may be stored at different locations such as locally on the client 165, on other systems in the LAN 125 or on other systems (not shown) connected to the global server 115.

The client 167 is similar to the client 165. However, workspace data stored on the client 167 is referred to as being stored in format B, which may be the same as or

6

different than format A. All aspects described above and below with reference to the client 165 are also possible with respect to the client 167. For example, client 167 may include services (not shown) accessible from remote terminal 105, may include a base system (not shown) for synchronizing workspace elements with the global server 115, etc.

The global server 115 includes a security system 160 for providing only an authorized user with secure access through firewalls to services. The security system 160 may perform identification and authentication services and may accordingly enable multiple levels of access based on the level of identification and authentication. The global server 115 further includes a configuration system 155 that downloads configuration data 356 (FIGS. 3 and 6) to the remote terminal 105 to configure remote terminal 105 components such as the operating system 270 (FIG. 2), the web engine 283 (FIG. 2), the applet engine 290 (FIG. 2), etc. The configuration system 155 uses the configuration data 356 to enable the remote terminal 105 to access the services provided by the service engine 175 and to provide a user-and-service-specific user interface.

The global server 115 stores workspace data 163, which includes an independently modifiable copy of each selected workspace element in the selected portions of the workspace data 180. Accordingly, the workspace data 163 includes an independently modifiable copy of each corresponding version information 782 (FIG. 7). The workspace data 163 may also include workspace elements which originate on the global server 115 such as e-mails sent directly to the global server 115 or workspace elements which are downloaded from another client (not shown). The global server 115 maintains the workspace data 163 in a format, referred to as a "global format," which is selected to be easily translatable by the global translator 150 to and from format A and to and from format B. As with format A and format B, one skilled in the art knows that the global format actually includes a global format for each information type. For example, there may be a global format for bookmarks (FIG. 5), a global format for files, a global format for calendar data, a global format for e-mails, etc.

The global server 115 also includes a synchronization agent 145 for examining the workspace elements of workspace data 163. More particularly, the base system 170 and the synchronization agent 145, collectively referred to herein as "synchronization means," cooperate to synchronize the workspace data 163 with the selected portions of the workspace data 180. The synchronization means may individually synchronize workspace elements (e.g., specific word processor documents) or may synchronize workspace element folders (e.g., a bookmark folder). Generally, the base system 170 manages the selected portions of the workspace data 180 within the LAN 125 and the synchronization agent 145 manages the selected portions of workspace data 163 within the global server 115. It will be appreciated that the global translator 150 cooperates with the synchronization means to translate between format A (or format B) and the global format. It will be further appreciated that the global server 115 may synchronize the workspace data 163 with workspace data 180 and with the workspace data (not shown) on the client 167. Accordingly, the workspace data 163 can be easily synchronized with the workspace data (not shown) on the client 167.

The remote terminal 105 includes a web engine 140, which sends requests to the global server 115 and receives information to display from the global server 115. The web engine 140 may use HyperText Transfer Protocol (HTTP)

and HyperText Markup Language (HTML) to interface with the global server 115. The web engine 140 may be enabled to run applets, which when executed operate as the security interface for providing access to the global server 115 and which operate as the application interface with the requested service. Using the present invention, a user can operate any remote client 105 connected to the Internet to access the global server 115, and thus to access the services and the workspace data on or accessible by the global server 115.

FIG. 2 is a block diagram illustrating details of the remote terminal 105, which includes a Central Processing Unit (CPU) 210 such as a Motorola Power PC™ microprocessor or an Intel Pentium™ microprocessor. An input device 220 such as a keyboard and mouse, and an output device 230 such as a Cathode Ray Tube (CRT) display are coupled via a signal bus 235 to CPU 210. A communications interface 240, a data storage device 250 such as Read Only Memory (ROM) and a magnetic disk, and a Random-Access Memory (RAM) 260 are further coupled via signal bus 235 to CPU 210. The communications interface 240 is coupled to a communications channel 110 as shown in FIG. 1.

An operating system 270 includes a program for controlling processing by CPU 210, and is typically stored in data storage device 250 and loaded into RAM 260 (as shown) for execution. Operating system 270 further includes a communications engine 275 for generating and transferring message packets via the communications interface 240 to and from the communications channel 110. Operating system 270 further includes an Operating System (OS) configuration module 278, which configures the operating system 270 based on OS configuration data 356 (FIG. 3) such as Transmission Control Protocol (TCP) data, Domain Name Server (DNS) addresses, etc. received from the global server 115.

Operating system 270 further includes the web engine 140 for communicating with the global server 115. The web engine 140 may include a web engine (WE) configuration module 286 for configuring elements of the web engine 140 such as home page addresses, bookmarks, caching data, user preferences, etc. based on the configuration data 356 received from the global server 115. The web engine 140 may also include an encryption engine 283 for using encryption techniques to communicate with the global server 115. The web engine 140 further may include an applet engine 290 for handling the execution of downloaded applets including applets for providing security. The applet engine 290 may include an Applet Engine (AE) configuration module 295 for configuring the elements of the applet engine 290 based on configuration data 356 received from the global server 115.

FIG. 3 is a block diagram illustrating details of the global server 115, which includes a Central Processing Unit (CPU) 310 such as a Motorola Power PC™ microprocessor or an Intel Pentium™ microprocessor. An input device 320 such as a keyboard and mouse, and an output device 330 such as a Cathode Ray Tube (CRT) display are coupled via a signal bus 335 to CPU 310. A communications interface 340, a data storage device 350 such as Read Only Memory (ROM) and a magnetic disk, and a Random-Access Memory (RAM) 370 are further coupled via signal bus 335 to CPU 310. As shown in FIG. 1, the communications interface 340 is coupled to the communications channel 110 and to the communications channel 120.

An operating system 380 includes a program for controlling processing by CPU 310, and is typically stored in data storage device 350 and loaded into RAM 370 (as illustrated)

for execution. The operating system 380 further includes a communications engine 382 for generating and transferring message packets via the communications interface 340 to and from the communications channel 345. The operating system 380 also includes a web page engine 398 for transmitting web page data 368 to the remote terminal 105, so that the remote terminal 105 can display a web page 900 (FIG. 9) listing functionality offered by the global server 115. Other web page data 368 may include information for displaying security method selections.

The operating system 380 may include an applet host engine 395 for transmitting applets to the remote terminal 105. A configuration engine 389 operates in conjunction with the applet host engine 395 for transmitting configuration applets 359 and configuration and user data 356 to the remote terminal 105. The remote terminal 105, executes the configuration applets 359 and uses the configuration and user data 356 to configure the elements (e.g., the operating system 270, the web engine 140 and the applet engine 290) of the remote terminal 105. Configuration and user data 356 is described in greater detail with reference to FIG. 6.

The operating system 380 also includes the synchronization agent 145 described with reference to FIG. 1. The synchronization agent 145 synchronizes the workspace data 163 on the global server 115 with the workspace data 180 on the client 165. As stated above with reference to FIG. 1, the global translator 150 translates between format A used by the client 165 and the global format used by the global server 115.

The operating system 380 may also include a security engine 392 for determining whether to instruct a communications engine 382 to create a secure communications link with a client 165 or terminal 105, and for determining the access rights of the user. For example, the security engine 392 forwards to the client 165 or remote terminal 105 security applets 362, which when executed by the receiver poll the user and respond back to the global server 115. The global server 115 can examine the response to identify and authenticate the user.

For example, when a client 165 attempts to access the global server 115, the security engine 384 determines whether the global server 115 accepts in-bound communications from a particular port. If so, the security engine 392 allows the communications engine 382 to open a communications channel 345 to the client 165. Otherwise, no channel will be opened. After a channel is opened, the security engine 392 forwards an authentication security applet 362 to the remote terminal 105 to poll the user for identification and authentication information such as for a user ID and a password. The authentication security applet 362 will generate and forward a response back to the global server 115, which will use the information to verify the identity of the user and provide access accordingly.

It will be appreciated that a "request-servicing engine" may be the configuration engine 389 and the applet host engine 395 when providing services to a remote terminal 105 or client 165. The request-servicing engine may be the web page engine 398 when performing workspace data 163 retrieval operations directly from the global server 115. The request-servicing engine may be the configuration engine 389 and the applet host engine 395 when performing workspace data 180 retrieval operations from the client 165 or from any other site connected to the global server 115. The request-servicing engine may be security engine 392 when performing security services such as user identification and authentication. The request-servicing engine may be the

synchronization agent when the performing synchronization with the client **165**. Further, the request-servicing engine may be any combination of these components.

FIG. 4 is a block diagram illustrating details of the synchronization agent **145**, which includes a communications module **405** and a general synchronization module **410**. The communications module **405** includes routines for compressing data and routines for communicating via the communications channel **120** with the base system **170**. The communications module **405** may further include routines for communicating securely channel through the global firewall **130** and through the LAN firewall **125**.

The general synchronization module **410** includes routines for determining whether workspace elements have been synchronized and routines for forwarding to the base system **170** version information (not shown) of elements determined to be modified after last synchronization. The general synchronization module **410** may either maintain its own last synchronization signature (not shown), receive a copy of the last synchronization signature with the request to synchronize from the base system **170**, or any other means for insuring that the workspace data has been synchronized. The general synchronization module **410** further includes routines for receiving preferred versions of workspace data **180** workspace elements from the base system **170**, and routines for forwarding preferred versions of workspace data **180** workspace elements to the base system **170**.

FIG. 5 illustrates an example bookmark workspace element in the global format. The translator **150** incorporates all the information needed to translate between all incorporated formats. For example, if for a first client a bookmark in format A needs elements X, Y and Z and for a second client a bookmark in format B needs elements W, X and Y, the global translator **150** incorporates elements W, X, Y and Z to generate a bookmark in the global format. Further, the translator **150** incorporates the information which is needed by the synchronization means (as described below in FIG. 4) such as the last modified date. Accordingly, a bookmark in the Global Format may include a user identification (ID) **505**, an entry ID **510**, a parent ID **515**, a folder ID flag **520**, a name **525**, a description **530**, the Uniform Resource Locator (URL) **535**, the position **540**, a deleted ID flag **545**, a last modified date **550**, a created date **555** and a separation ID flag **560**.

FIG. 6 is a block diagram illustrating details of the configuration and user data **356**. Configuration data **356** includes settings **605** such as TCP data and the DNS address, web browser settings such as home page address, bookmarks and caching data, applet engine settings, and applet configuration data such as the user's e-mail address name and signature block. It will be appreciated that applet-specific configuration and user data **356** is needed, since the service may not be located on the user's own local client **165**. Configuration and user data **356** further includes predetermined user preferences **610** such as font, window size, text size, etc.

Configuration data **356** further includes the set of services **615**, which will be provided to the user. Services **615** include a list of registered users and each user's list of user-preferred available services **615**. Services may also include a list of authentication levels needed to access the services **615**. Configuration and user data **137** further includes service addresses **620** specifying the location of each of the services **615** accessible via the global server **115**.

FIG. 7 is a block diagram illustrating details of the client **165**, which includes a CPU **705**, an input device **710**, an

output device **725**, a communications interface **710**, a data storage device **720** and RAM **730**, each coupled to a signal bus **740**.

An operating system **735** includes a program for controlling processing by the CPU **705**, and is typically stored in the data storage device **720** and loaded into the RAM **730** (as illustrated) for execution. A service engine **175** includes a service program for managing workspace data **180** that includes version information (not shown). The service engine **175** may be also stored in the data storage device **720** and loaded into the RAM **730** (as illustrated) for execution. The workspace data **180** may be stored in the data storage device **330**. As stated above with reference to FIG. 1, the base system **170** operates to synchronize the workspace data **180** on the client **165** with the workspace data **163** on the global server **115**. The base system **170** may be also stored in the data storage device **720** and loaded into the RAM **730** (as shown) for execution. The base system **170** is described in greater detail with reference to FIG. 8.

FIG. 8 is a block diagram illustrating details of the base system **170**, which includes a communications module **805**, a user interface module **810**, locator modules **815**, a synchronization-start ("synch-start") module **820**, a general synchronization module **825** and a content-based synchronization module **830**. For simplicity, each module is illustrated as communicating with one another via a signal bus **840**. It will be appreciated that the base system **170** includes the same components as included in the synchronization agent **145**.

The communications module **805** includes routines for compressing data, and routines for communicating via the communications interface **710** (FIG. 7) with the synchronization agent **145** (FIG. 1). The communications module **805** may include routines for applying Secure Socket Layer (SSL) technology and user identification and authentication techniques (i.e., digital certificates) to establish a secure communication channel through the LAN firewall **135** and through the global firewall **130**. Because synchronization is initiated from within the LAN firewall **135** and uses commonly enabled protocols such as HyperText Transfer Protocol (HTTP), the typical firewall **135** which prevents in-bound communications in general and some outbound protocols does not act as an impediment to e-mail synchronization. Examples of communications modules **805** may include TCP/IP stacks or the AppleTalk™ protocol.

The user interface **810** includes routines for communicating with a user, and may include a conventional Graphical User Interface (GUI). The user interface **810** operates in coordination with the client **165** components as described herein.

The locator modules **815** include routines for identifying the memory locations of the workspace elements in the workspace data **180** and the memory locations of the workspace elements in the workspace data **163**. Workspace element memory location identification may be implemented using intelligent software, i.e., preset memory addresses or the system's registry, or using dialogue boxes to query a user. It will be appreciated that the locator modules **815** may perform workspace element memory location identification upon system boot-up or after each communication with the global server **115** to maintain updated memory locations of workspace elements.

The synchronization-start module **820** includes routines for determining when to initiate synchronization of workspace data **163** and workspace data **180**. For example, the synchronization-start module **820** may initiate data synchro-

11

nization upon user request, at a particular time of day, after a predetermined time period passes, after a predetermined number of changes, after a user action such as user log-off or upon like criteria. The synchronization-start module 820 initiates data synchronization by instructing the general synchronization module 825 to begin execution of its routines. It will be appreciated that communications with synchronization agent 145 preferably initiate from within the LAN 125, because the typical LAN firewall 125 prevents in-bound communications and allows out-bound communications.

The general synchronization module 825 includes routines for requesting version information from the synchronization agent 145 (FIG. 1) and routines for comparing the version information against a last synchronization signature 835 such as a last synchronization date and time to determine which versions have been modified. The general synchronization module 825 further includes routines for comparing the local and remote versions to determine if only one or both versions of a particular workspace element have been modified and routines for performing an appropriate synchronizing responsive action. Appropriate synchronizing responsive actions may include forwarding the modified version (as the preferred version) of a workspace element in workspace data 180 or forwarding just a compilation of the changes to the other store(s). Other appropriate synchronizing responsive actions may include, if reconciliation between two modified versions is needed, then instructing the content-based synchronization module 830 to execute its routines (described below).

It will be appreciated that the synchronization agent 145 preferably examines the local version information 124 and forwards only the elements that have been modified since the last synchronization signature 835. This technique makes efficient use of processor power and avoids transferring unnecessary data across the communications channel 712. The general synchronization module 825 in the LAN 135 accordingly compares the data elements to determine if reconciliation is needed. Upon completion of the data synchronization, the general synchronization module 825 updates the last synchronization signature 835.

The content-based synchronization module 830 includes routines for reconciling two or more modified versions of workspace data 163, 180 in the same workspace element. For example, if the original and the copy of a user workspace element have both been modified independently since the last synchronization, the content-based synchronization module 830 determines the appropriate responsive action. The content-based synchronization module 830 may request a user to select the preferred one of the modified versions or may respond based on preset preferences, i.e., by storing both versions in both stores or by integrating the changes into a single preferred version which replaces each modified version at both stores. When both versions are stored at both stores, each version may include a link to the other version so that the user may be advised to select the preferred version.

It will be appreciated that any client 165 that wants synchronization may have a base system 170. Alternatively, one base system 170 can manage multiple clients 165. It will be further appreciated that for a thin client 165 of limited computing power such as a smart telephone, all synchronization may be performed by the global server 115. Accordingly, components of the base system 170 such as the user interface module 810, the locator modules 815, the general synchronization module 825 and the content-based synchronization module 830 may be located on the global

12

server 115. To initiate synchronization from the client 165, the client 165 includes the communications module 805 and the synch-start module 820.

FIG. 9 illustrates an example list 900 of accessible services provided by a URL-addressable HyperText Markup Language (HTML)-based web page, as maintained by the web page engine 398 of the global server 115. The list 900 includes a title 910 "Remote User's Home Page," a listing of the provided services 615 and a pointer 970 for selecting one of the provided services 615. As illustrated, the provided services may include an e-mail service 920, a calendaring service 930, an internet access service 940, a paging service 950, a fax sending service 960, a user authentication service 963 and a workspace data retrieval service 967. Although not shown, other services 615 such as bookmarking, QuickCard™, etc. may be included in the list 900. Although the web page provides the services 615 in a list 900, other data structures such as a pie chart or table may alternatively be used.

FIG. 10 is a flowchart illustrating a method 1000 for enabling a user to access the services 615 in the computer network system 100. Method 1000 begins by the remote terminal 105 in step 1005 creating a communications link with the global server 115. The global server 115 in step 1010 confirms that the user has privileges to access the functionality of the global server 115. Confirming user access privileges may include examining a user certificate, obtaining a secret password, using digital signature technology, performing a challenge/response technique, etc. It will be appreciated that the security engine 392 may cause the applet host engine 395 to forward via the communications channel 345 to the remote terminal 105 an authentication security applet 362 which when executed communicates with the global server 115 to authenticate the user.

After user access privileges are confirmed, the web page engine 398 of the global server 115 in step 1015 transmits web page data 368 and configuration and user data 356 to the remote terminal 105. The web engine 140 of the remote terminal 105 in step 1020 uses the web page data 368 and the configuration and user data 356 to display a web page service list 900 (FIG. 9) on the output device 230, and to enable access to the services 615 which the global server 115 offers. An example service list 900 is shown and described with reference to FIG. 9. Configuration of the remote terminal 105 and of the web page 700 is described in detail in the cross-referenced patent applications.

From the options listed on the web page 900, the user in step 1025 selects a service 615 via input device 220. In response, the request-servicing engine (described with reference to FIG. 3) provides the selected service 615. For example, the applet host engine 395 of the global server 115 in step 1030 may download to the remote terminal 105 a corresponding applet 359 and configuration and user data 356 for executing the requested service 615. Alternatively, the web page engine 398 may use, for example, HTTP and HTML to provide the selected service 615. As described above with reference to FIG. 6, the configuration and user data 356 may include user-specific preferences such as user-preferred fonts for configuring the selected service 615. Configuration and user data 356 may also include user-specific and service-specific information such as stored bookmarks, calendar data, pager numbers, etc. Alternatively, the corresponding applet 359 and the configuration and user data 356 could have been downloaded in step 1015. Providing access to the service by an applet 359 is described in greater detail below with reference to FIGS. 12-14.

The applet engine 290 of the remote terminal 105 in step 1035 initiates execution of the corresponding downloaded

13

applet. The global server 115 in step 1040 initiates the selected service 615 and in step 1045 selects one of three modes described with reference to FIGS. 12–14 for accessing the service 615. For example, if the user selects a service 615 on a service server (e.g., the client 165) that is not protected by a separate firewall, then the global server 115 may provide the user with direct access. If the user selects a service 615 provided by a service server within the LAN 125, then the global server 115 may access the service 615 as a proxy for the user. It will be appreciated that each firewall 130 and 135 may store policies establishing the proper mode of access the global server 115 should select. Other factors for selecting mode of access may include user preference, availability and feasibility. The global server 115 in step 1050 uses the selected mode to provide the remote terminal 105 user with access to the selected service 615.

FIG. 11 is a flowchart illustrating details of step 1005, which begins by the remote terminal 105 in step 1105 using a known Uniform Resource Locator (URL) to call the global server 115. The global server 115 and the remote terminal 105 in step 1107 create a secure communications channel therebetween, possibly by applying Secure Sockets Layer (SSL) technology. That is, the security engine 392 of the global server 115 in step 1110 determines if in-bound secure communications are permitted and, if so, creates a communications channel with the remote terminal 105. The web engine 140 of the remote terminal 105 and the security engine 392 of the global server 115 in step 1115 negotiate secure communications channel parameters, possibly using public key certificates. An example secure communications channel is RSA with RC4 encryption. Step 1115 thus may include selecting an encryption protocol which is known by both the global server 115 and the remote terminal 105. The encryption engine 283 of the remote terminal 105 and secure communications engine 392 of the global server 115 in step 1120 use the secure channel parameters to create the secure communications channel. Method 505 then ends.

FIG. 12 is a flowchart illustrating details of step 1050 in a first embodiment, referred to as step 1050a, wherein the global server 115 provides the remote terminal 105 with a direct connection to a service 615. Step 1050a begins by the applet engine 290 in step 1205 running a configuration applet 359 for the selected service 615 that retrieves the service address 620 from data storage device 380 and the authentication information from the keystore 365. The communications interface 340 in step 1210 creates a direct and secure connection with the communications interface 340 of the global server 115 at the retrieved service address 620, and uses the authentication information to authenticate itself. The applet in step 1215 acts as the I/O interface with the service 615. Step 1050a then ends.

FIG. 13 is a flowchart illustrating details of step 1050 in a second embodiment, referred to as step 1050b, wherein the global server 115 acts for the remote terminal 105 as a proxy to the service 615. Step 1050b begins with a configuration applet 359 in step 1305 requesting the service address 620 for the selected service 615, which results in retrieving the service address 620 directing the applet 359 to the global server 115. The applet 359 in step 1310 creates a connection with communications interface 340 of the global server 115. The global server 115 in step 1315 retrieves the service address 620 of the selected service 615 and the authentication information for the selected service 615 from the keystore 365. The communications interface 340 of the global server 115 in step 1320 negotiates secure channel parameters for creating a secure channel with the service server 1014. The communications interface 340 in step 1320 also authenticates itself as the user.

14

Thereafter, the applet 359 in step 1325 acts as the I/O interface with the communications interface 340 of the global server 115. If the global server 115 in step 1330 determines that it is unauthorized to perform a remote terminal 105 user's request, then the global server 115 in step 1345 determines whether the method 1050b ends, e.g., whether the user has quit. If so, then method 1050b ends. Otherwise, method 1050b returns to step 1325 to obtain another request. If the global server 115 in step 1330 determines that it is authorized to perform the remote terminal 105 user's request, then the global server 115 in step 1340 acts as the proxy for the remote terminal 105 to the service 615. As proxy, the global server 115 forwards the service request to the selected service 615 and forwards responses to the requesting applet 359 currently executing on the remote terminal 105. Method 1050b then jumps to step 1345.

FIG. 14 is a flowchart illustrating details of step 1050 in a third embodiment, referred to as step 1050c, wherein the service 615 being requested is located on the global server 115. Step 1050 begins with an applet in step 1405 retrieving the service address 620 for the selected service 615, which results in providing the configuration applet 359 with the service address 620 of the service 615 on the global server 115. Thus, the applet in step 1410 creates a secure connection with the global server 115. No additional step of identification and authentication is needed since the remote terminal 105 has already identified and authenticated itself to the global server 115 as described with reference to step 1010 of FIG. 10.

In step 1415, a determination is made whether the service 615 is currently running. If so, then in step 1425 a determination is made whether the service 615 can handle multiple users. If so, then the global server 115 in step 1430 creates an instance for the user, and the applet in step 1440 acts as the I/O interface with the service 615 on the global server 115. Method 1050c then ends. Otherwise, if the service 615 in step 1425 determines that it cannot handle multiple users, then method 1050c proceeds to step 1440. Further, if in step 1415 the global server 115 determines that the service 615 is not currently running, then the global server 115 in step 1420 initializes the service 615 and proceeds to step 1425.

FIG. 15 is a flowchart illustrating a method 1500 for using a global translator 150 to synchronize workspace data 163 and workspace data 180 in a secure network 100. Method 1500 begins with the user interface 900 in step 1505 enabling a user to select workspace elements of workspace data 163 and workspace data 180 for the synchronization means to synchronize. The locator modules 815 in step 1510 identify the memory locations of the workspace elements in workspace data 163 and workspace data 180. If a selected workspace element does not have a corresponding memory location, such as in the case of adding new workspace elements to the global server 115, then one is selected. The selected memory location may be a preexisting workspace element or a new workspace element. As stated above, workspace element memory location identification may be implemented using intelligent software or dialogue boxes. The general synchronization module 825 in step 1515 sets the previous status of the workspace elements equal to the null set, which indicates that all information of the workspace element has been added.

The synchronization-start module 820 in step 1520 determines whether predetermined criteria have been met which indicate that synchronization of the workspace elements selected in step 1505 should start. If not, then the

15

synchronization-start module **820** in step **1525** waits and loops back to step **1520**. Otherwise, the communications module **805** and the communications module **405** in step **1530** establish a secure communications channel therebetween.

The general synchronization module **825** in step **1535** determines whether any workspace elements have been modified. That is, the general synchronization module **825** in step **1535** examines the version information of each selected workspace element in the workspace data **180** against the last synchronization signature **435** to locate modified workspace elements. This comparison may include comparing the date of last modification with the date of last synchronization, or may include a comparison between the current status and the previous status as of the last interaction. Similarly, the general synchronization module **815** examines the version information of each corresponding workspace element in workspace data **163** and the last synchronization signature **435** to locate modified workspace elements.

If in step **1535** no modified workspace elements or folders are located, then the general synchronization module **825** in step **1560** updates the last synchronization signature **435** and method **1500** ends. Otherwise, the general synchronization module **825** in step **1540** determines whether more than one version of a workspace element has been modified since the last synchronization.

If only one version has been modified, then the corresponding general synchronization module **825** in step **1545** determines the changes made. As stated above, determining the changes made may be implemented by comparing the current status of the workspace element against the previous status of the workspace element as of the last interaction therebetween. If the changes were made only to the version in the workspace data **163**, then the global translator **150** in step **1550** translates the changes to the format used by the other store, and the general synchronization module **410** in step **1555** forwards the translated changes to the general synchronization module **825** for updating the outdated workspace element in the workspace data **180**. If the updated version is a workspace element in the workspace data **180**, then the general synchronization module **825** sends the changes to the updated version to the global translator **150** for translation and then to the general synchronization module **410** for updating the outdated workspace element in the workspace data **163**. The general synchronization module **825** and the general synchronization module **410** in step **1557** update the previous state of the workspace element to reflect the current state as of this interaction. Method **1500** then returns to step **1535**.

If the general synchronization module **825** in step **1540** determines that multiple versions have been modified, then the general synchronization module **825** in step **1565** computes the changes to each version and in step **1570** instructs the content-based synchronization module **830** to examine content to determine if any conflicts exist. For example, the content-based synchronization module **830** may determine that a conflict exists if a user deletes a paragraph in one version and modified the same paragraph in another version. The content-based synchronization module **830** may determine that a conflict does not exist if a user deletes different paragraphs in each version. If no conflict is found, then method **1500** jumps to step **1550** for translating and forwarding the changes in each version to the other store. However, if a conflict is found, then the content-based synchronization module **830** in step **1575** reconciles the modified versions. As stated above, reconciliation may

16

include requesting instructions from the user or based on previously selected preferences performing responsive actions such as storing both versions at both stores. It will be appreciated that a link between two versions may be placed in each of the two versions, so that the user will recognize to examine both versions to select the preferred version. Method **1500** then proceeds to step **1550**.

It will be further appreciated that in step **1510** new workspace elements and preexisting workspace elements to which new workspace elements will be merged are set to "modified" and the previous status is set to the null set. Thus, the general synchronization module **825** in step **1540** will determine that more than one version has been modified and the content-based synchronization module **830** in step **1570** will determine that no conflict exists. The changes in each will be translated and forwarded to the other store. Accordingly, the two versions will be effectively merged and stored at each store.

For example, if a first bookmark folder was created by the web engine **140** on the client **165**, a second folder was created by a web engine **140** on the remote terminal **105**, no preexisting folder existed on the global server **115** and the user selected each of these folders for synchronization, then the synchronization means will effectively merge the first and second folders. That is, the general synchronization module **825** on the client **165** will determine that the first folder has been modified and the previous status is equal to the null set. The general synchronization module **825** will determine and send the changes, i.e., all the workspace elements in the first folder, to a new global folder on the global server **115**. Similarly, the general synchronization module (not shown) on the remote terminal **105** will determine that, as of its last interaction, the previous status of each of the second and the global folders is the null set. The general synchronization module **825** will instruct the content-based synchronization module **830** to examine the changes made to each folder to determine whether a conflict exists. Since no conflicts will exist, the general synchronization module **825** will forward the changes to the global folder and the general synchronization module **410** will forward its changes to the second store, thereby merging the workspace elements of the first and second folders in the global and second folders. The general synchronization module **410** will inform the general synchronization module **825** that the global folder has been modified relative to the last interaction, and will forward the new changes to the first folder. Thus, the first and second folders will be merged and stored at each store.

The foregoing description of the preferred embodiments of the invention is by way of example only, and other variations of the above-described embodiments and methods are provided by the present invention. For example, a server can be any computer which is polled by a client. Thus, the remote terminal **105** may be referred to as a type of client. Although the system and method have been described with reference to applets, other downloadable executables such as Java™ applets, Java™ applications or ActiveX™ control developed by the Microsoft Corporation can alternatively be used. Components of this invention may be implemented using a programmed general-purpose digital computer, using application specific integrated circuits, or using a network of interconnected conventional components and circuits. The embodiments described herein have been presented for purposes of illustration and are not intended to be exhaustive or limiting. Many variations and modifications are possible in light of the foregoing teaching. The invention is limited only by the following claims.

17

What is claimed is:

1. A method for synchronizing workspace data, comprising:
 storing first workspace data on a first device;
 storing second workspace data on a second device;
 determining differences between the first workspace data
 and the second workspace data;
 storing the differences at a global server; and
 sending the differences from the global server to the
 second device.
2. The method of claim 1, wherein the first workspace
 data comprise a workspace data element from a first user of
 the first device to a second user of the second device.
3. The method of claim 2, wherein the workspace data
 element includes data selected from a group including email
 data, file data, calendar data, user data and bookmark data.
4. The method of claim 1, wherein at least one of the first
 device and the second device is selected from a group
 including a smart phone, a television settop box and a
 personal computer.
5. The method of claim 1, further comprising continuing
 to store the differences at the global server is continued after
 the sending.
6. The method of claim 1, further comprising storing at
 the server version-indicating information corresponding to
 the differences.
7. The method of claim 1, further comprising merging, by
 the second device, the differences with third workspace data
 stored on the second device.
8. A system for synchronizing workspace data, compris-
 ing:

18

- means for storing first workspace data on a first device;
 means for storing second workspace data on a second
 device;
 means for determining differences between the first work-
 space data and the second workspace data;
 means for storing the differences at a global server; and
 means for sending the differences from the global server
 to the second device.
9. The system of claim 8, wherein the first workspace data
 comprises a workspace data element from a first user of the
 first device to a second user of the second device.
 10. The system of claim 9, wherein the workspace data
 element includes data selected from a group including email
 data, file data, calendar data, user data and bookmark data.
 11. The system of claim 8, wherein at least one of the first
 device and the second device is selected from a group
 including a smart phone, a television settop box and a
 personal computer.
 12. The system of claim 8, further comprising means for
 continuing to store the differences at the global server is
 continued after the sending.
 13. The system of claim 8, further comprising means for
 storing at the server version-indicating information corre-
 sponding to the differences.
 14. The system of claim 8, further comprising means for
 merging, by the second device, the differences with third
 workspace data stored on the second device.

* * * * *